System Design Document for Wack

Viktor Franzen, Tobias Lindroth, Spondon Siddiqui, Alexander Solberg

October 11, 2018

1 Introduction

This document is supposed to give the reader an deeper understanding about the design of our project, "Wack".

1.1 Design goals

The main design goal of this project is to have an easily extended application that is loosely coupled. It should be easily tested and the application should also have a clear MVC-structure.

1.2 Definitions, acronyms, and abbreviations

GUI: Graphical User Interface; How the application looks.

MVC: Model-View-Controller; A design pattern that separates the logic from the view which makes it easy to reuse the model or change view.

UML: Unified Modeling Language; A genereal-purpose, modeling language used in the development of object oriented projects. It provides a standard way for the design of a system to be visualized.

2 System Architecture

Everything will be on the same computer, but using multiple clients. This will simulate an actual chat application with multiple computers.

During the planning stage of this application it was determined that a MVC pattern would be used. Therefore, during the implementation phase of the project, the application was split into a model, a view, and a controller. This means that the model is not aware of what entity is using its data and logic.

Additionally, the separate views have a relation with its given model but are completely unaware of the controllers. The controllers act as a connection between the model and its respective view. In this implementation the controller interacts with a facade towards the model and uses the information it receives to update the views. The important factor in the MVC implementation is that either package can be replaced without it affecting the program.

2.1 Subsystem Decomposition

Our chat application is divided into 3 major packages which can be seen in "figure x". See the appendix to see the general relationships between the packages and a fully expanded image of the packages.

Model contains all the data and logic which is used to run the application.

View accepts user input and then renders the view based on the models updated data.

Controller handles inputs and requests from view and communicates with the model.

2.2 Model

2.3 'Second Component'

3 Persistent Data Management

4 Access Control and Security

In the application there is different kind of roles which changes some of the things a user can do in the application.

User: The most basic role anyone using the application can have. As soon as you login you become a user. The only thing a user can do is join channels and see other channels.

Channel member: As soon as you join a channel you become a channel member. As a channel member you can write and read messages that are written inside the channel. You also can add other members to the channel.

Channel administrator: If you create a channel you automatically become a channel administrator. You can also be appointed the role by another channel administrator. A channel administrator can change the channel image, kick members from the channel and also all the functionality a ordinary channel member.

Message sender: A message sender is the user that sent a message. This is stored in the message as a reference to the user that created it.

5 References